IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Serial No.: 10/691,109	Group Art Unit: 2141
Filed: October 20, 2003	Examiner: Brian J Gillis
Applicant: Alex Conta	Attorney Docket: TRA-084

Title: Methods and Apparatus for Implementing Multiple Types of Network Tunneling in a Uniform Manner

Commissioner for Patents
Alexandria, VA 22313

Sir:

APPLICANT'S BRIEF ON APPEAL

On September 4, 2007 all claims (i.e. claims 1-9 and 14-40) were finally rejected by the Examiner. On October 11, 2007, the Applicant filed a Notice of Appeal (with fee) appealing the final rejection of claims 1-9 and 14-40. Please charge the $255 small entity appeal brief fee, and any additional fee which may be due, to deposit account no. 07-1732.

TABLE OF CONTENTS

I. Real Party in Interest

The real party in interest is Transwitch Corporation by virtue of an assignment recorded at Reel: 014660, Frame: 0234 on October 21, 2003.

II. Related Appeals and Interferences

There are no other appeals or interferences which are related to this application or this appeal.

III. Status of the Claims

This application was filed with claims 1-40. On February 7, 2007, claim 10 was canceled and claims 11-13 were withdrawn from consideration.

The final rejection of claims 1-9 and 14-40 was appealed on October 11, 2007. Therefore, claims 1-9 and 14-40 are the claims on appeal.

IV. Status of Amendments

All amendments have been entered. No amendments have been filed after final rejection.

V. Summary of the Claimed Subject Matter

Independent claim 1 is directed to a uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input interfaces and a plurality of output interfaces. The method includes providing a finite set of tunnel interfaces (e.g. T1, T2,..., Tm, in Fig. 1 or 20, 22, 24, 26 in Fig. 2, page 14, lines 3-19) each characterized by a set of tunnel specific attributes (e.g. lines 82-94 of the Evidence Appendix illustrate attributes of an IPv4 tunnel interface and lines 114-128 illustrate attributes of an MPLS tunnel interface, page 25, lines 1-24). A first of the tunnel interfaces (e.g. 20 in Fig. 2) is associated with one tunneling protocol (e.g. IP in IP in Fig. 2) and a second tunnel interface (e.g. 26 in Fig. 2) is associated with a second tunneling protocol (e.g. MPLS in Fig. 2) different from the first protocol (page 15, line 12 through page 18, line 6). The method of claim 1 further includes mapping one of the input interfaces (e.g. 10, 12, 14 in Fig. 2) to one of the tunnel interfaces (e.g. 20, 22, 24, 26 in Fig. 2) and mapping the one of the tunnel interfaces (e.g. 20, 22, 24, 26 in Fig. 2) to one of the output interfaces (30, 32, 34, 36 in Fig. 2, page15, line 12 through page 18, line 6).

Independent claim 9 is directed to a uniform method for implementing multiple tunneling protocols in a switch or router. The method includes associating an input interface (e.g. 10, 12, 14 in Fig. 2, page 15, line 12 through page 18, line 6), an output interface (e.g. 30, 32, 34, 36 in Fig. 2), and an information database (e.g. FIB 210, TSIB 212, SIB 214 in Fig. 2) with each of the multiple tunneling protocols (e.g. IP in IP, MPLS, L2TP in Fig. 2) . The method also includes associating a mapping interface (e.g.

encapsulation engines 21, 23, 25 in Fig. 2) and a mapping information base (e.g. FIB 220, 222 in Fig. 2) with each of the multiple tunneling protocols (e.g. IP in IP, MPLS, L2TP in Fig. 2). The method further includes uniformly implementing a tunneling protocol by selecting an input interface, an output interface, and an information database associated with the tunneling protocol to be implemented (page 12, lines 10-14, page 18, lines 11-21).

Independent claim 15 is directed to a uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input streams and a plurality of output streams. The method includes providing a finite set of tunnel interfaces (e.g. 20, 22, 24, 26 in Fig. 2, page 14, lines 3-19) where a first tunnel interface is associated with one tunneling protocol (e.g. 20 with IP in IP in Fig. 2) and a second tunnel interface is associated with a second tunneling protocol different from the first tunneling protocol (e.g. 26 with MPLS in Fig. 2, page 15, line 12 through page 18, line 6). The method further includes mapping input streams (e.g. through input interfaces 10, 12, 14 in Fig. 2) and output streams (e.g. through output interfaces 30, 32, 34, 36 in Fig. 2, page 15, line 12 through page 18, line 6) to tunnel interfaces (e.g. 20, 22, 24, 26 in Fig. 2) for different tunneling protocols (e.g. IP in IP, MPLS, L2TP in Fig. 2) in a uniform manner.

Independent claim 22 is directed to a uniform method for implementing multiple tunneling protocols in a switch or router. The method includes providing a plurality of tunnel interfaces (e.g. 20, 22, 24, 26 in Fig. 2, page 14, lines 3-19) where a first tunnel interface is associated with one tunneling protocol (e.g. 20 with IP in IP in Fig. 2) and a second tunnel interface is associated with a second tunneling protocol different from the first tunneling protocol(e.g. 26 with MPLS in Fig. 2, page 15, line 12 through page 18, line 6), each tunnel interface having a plurality of parameters which are described in a uniform way, the plurality of parameters including a local source address and a remote destination address (page 23, line 14 through page 24, line 14, e.g. lines 45-49, 90-92, 161-193 of the Evidence Appendix).

Independent claim 31 is directed to an application programming interface (API) for implementing a plurality of different tunneling protocols in a switch or router. See, generally the Evidence Appendix and Specification page 24, line 16 through page 26, line 4. The API includes a tunneling interface data structure (e.g. page 24, lines 16 through 25) having a plurality of parameters; and a plurality of functions (e.g. lines 182-180, 192-201, 211-222, 225-237 of the Evidence Appendix) for setting the parameters of the tunneling interface data structure, wherein a tunneling interface data structure is configurable to implement any one of the plurality of different tunneling protocols by using at least some of the plurality of functions.

VI. Grounds of Rejection to be Reviewed on Appeal

Thirteen issues are presented on appeal:

a) whether claims 31, 32, and 36 are anticipated under 35 U.S.C. §102(e) by Shrader;

b) whether claims 1-8, 15, 22, and 25 are obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves;

c) whether claims 9 and 35 are obvious under 35 U.S.C. §103(a) over Shrader in view of Hauck;

d) whether claim 14 is obvious under 35 U.S.C. §103(a) over Shrader in view of Hauck and further in view of admitted prior art;

e) whether claims 16-21, 23, and 28 are obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves and further in view of Miller;

f) whether claim 24 is obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves in view of Miller and further in view of Rekhter;

g) whether claim 26 is obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves and further in view of Shrader;

h) whether claim 27 is obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves in view of Shrader and further in view of Tsirtsis;

i) whether claims 29 and 30 are obvious under 35 U.S.C. §103(a) over Hauck in view of Greaves and further in view of admitted prior art;

j) whether claims 33 and 38 are obvious under 35 U.S.C. §103(a) over Shrader in view of Miller;

k) whether claim 34 is obvious over under 35 U.S.C. §103(a) Shrader in view of Miller and further in view of Rekhter;

l) whether claim 37 is obvious under 35 U.S.C. §103(a) over Shrader in view of Tsirtsis; and

m) whether claims 39 and 40 are obvious under 35 U.S.C. §103(a) over Shrader in view of admitted prior art.

VII. Argument

The argument is presented in the same order as the rejections are set forth in the Final Office Action.

a) Claims 31, 32, and 36 stand rejected under 35 U.S.C. §102(e) as anticipated by Shrader.

Independent claim 31 is directed to an application programming interface (API) for implementing a plurality of different protocols in a switch or router. The API includes a tunneling interface data structure which is configurable to implement any one of a plurality of ***different tunneling protocols***. Schrader shows a graphical interface for managing IP tunnels between two firewalls. All of the tunnels use the same tunneling protocol, i.e. IP over IP. The individual tunnels can be configured, e.g. to filter packets coming from a certain IP address, to employ encryption, etc., but multiple tunneling protocols can not be achieved in a single switch through Shrader's API. The only mention of protocol in Shrader is at column 16, lines 44 – 53 which are reproduced below:

> "While the description above has been related to IP tunneling and filtering, i.e. the tunnel and filter rules promulgated by the various Internet bodies, the invention has application to ***any set of tunnel and filter rules*** which may be imposed between secure and nonsecure networks. For example, ***the point to point tunneling protocol*** originally proposed by the Microsoft Corporation ***could be***

administrated by the use of the present invention with a minimum of adaption." [Emphasis added.]

There is no suggestion in Shrader of implementing a plurality of tunneling protocols. A fair reading of the cited portion of Shrader suggests that any ***one*** set of tunnel and filter rules might be implemented. Claim 31 provides that the "tunneling interface data structure is configurable to implement any one of said plurality of different tunneling protocols by using at least some of said plurality of functions."

In addition, while Shrader states that his system can be adapted to a different protocol, he provides no teaching of how to accomplish this and clearly does not suggest implementing multiple tunneling protocols in the same switch or router.

In view of the foregoing, it is submitted that claim 31 and its dependents are allowable over Shrader.

b) Claims 1-8, 15, 22 and 25 stand rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves.

Claim 1 was previously amended to make it more clear that the invention implements ***multiple different*** tunneling protocols. As described in the instant specification, tunneling is a process whereby a data packet is encapsulated in another packet before traversing a network. There are two main reasons for tunneling. One is to transport one type of packet over a network designed for another type of packet, e.g.

Ethernet over ATM. Another application for tunneling is to create a Virtual Private Network, a process whereby a secure connection is created across a public network through the use of tunneling.

Currently there are a wide variety of tunneling protocols. Among the most popular protocols are: IP (Internet Protocol) over IP, IP over MPLS (Multiprotocol Label Switching), Ethernet over MPLS, and L2TP (Layer 2 Tunneling Protocol). Each of these tunneling protocols requires different processing of packets. See pages 1-9 of the instant specification which explains how each tunneling protocol is presently implemented. The present invention provides a uniform method for implementing all of these different protocols in a single switch or router.

Hauck describes a system and method for network tunneling utilizing micro-flow state information. A micro-flow is a uniquely identifiable data stream. In Hauck, all of the micro-flows are IP data streams. Each micro-flow may utilize a different IP protocol such as TCP (Transmission Control Protocol), UDP (User Datagram Protocol), FTP (File Transfer Protocol), etc. It should be noted that TCP, UDP, and FTP are not tunneling protocols. According to Hauck, micro-flows are aggregated into a flow block and flow blocks are assigned to tunnels. All of the tunnels in Hauck are MPLS tunnels. Hauck does not teach how to implement any tunneling protocol. Hauck is concerned with managing micro-flows.

In sum, Hauck discloses IP data arriving at a router, aggregating flow blocks and sending micro-flows through an MPLS tunnel to another router where they are extracted. ***Only one tunneling protocol is used in Hauck***. Hauck's primary concern is to implement QoS (quality of service) through the use of micro-flows and aggregate flow blocks. Implementing different tunneling protocols is not an issue in Hauck.

In the Final Rejection, the Examiner has admitted that Hauck does not teach multiple tunneling protocols and relies on Greaves to teach this. Greaves is not concerned with tunneling, is directed to non-analogous art, and actually teaches away from the claimed invention. The relevant part of §103(a) states:

> "A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between ***the subject matter sought to be patented*** and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill ***in the art to which said subject matter pertains***." [Emphasis added.]

The Examiner argues that Greaves and Hauck are analogous art because they are both related to data transfer. It is respectfully submitted that the field of data transfer encompasses many non-analogous arts. Moreover, the present invention is directed to "tunneling" which assumes that the data transfer issues are already taken care of by the switch or router. That is, the subject matter sought to be patented is not the data transfer which will be provided by the switch or router; it is tunneling. What is sought to be patented is a uniform method for implementing multiple tunneling protocols.

The "protocols" discussed by Greaves, which are generic data transfer protocols such as HTTP, are so different from the protocols in the present invention that Greaves actually teaches away from the claimed invention. Paragraph 4 of Greaves is reproduced below:

> "A protocol operates over an interface. Interfaces occur where sections touch each other. Any particular section has one or more interfaces. There can be more than one interface between any given pair of sections. ***Each interface operates its protocol without reference to the other interfaces.*** Source and sink sections are used frequently in system test benches: these have a single interface and just generate or soak up data respectively." [Emphasis added.]

In contrast, it is clear from the claims of the present invention that the protocols implemented by the invention reference both the input interface and the output interface. Claim 1 provides that tunneling protocols are operated in part by "b) mapping one of the ***input interfaces*** to one of said tunnel interfaces; and c) mapping said one of said tunnel interfaces to one of the ***output interfaces***...". [Emphasis added.] Thus, the tunneling protocols require reference to two interfaces. The Examiner's rejection illustrates the flaws in patent searching with key words. It is common that identical words will have different meanings in different arts. Thus, the "protocols" to which Greaves is directed are "generic data transfer protocols" such as HTTP. "Like any generic data transfer protocol, HTTP cannot regulate the content of the data that is transferred, nor is there any *a priori* method of determining the sensitivity of any particular piece of information within the context of any given request." See, HTTP/1.0, v10-spec-05, T. Berners-Lee, et al., February 19, 1996. On the other hand, the protocols of the present invention are tunneling protocols which clearly regulate the content of the data that is transferred, for example by the alteration of headers, etc. Thus, the combination of Greaves and Hauck does not suggest the use of multiple tunneling protocols in a single switch or router, nor does it suggest implementing multiple tunneling protocols in a uniform manner, and the Examiner's obviousness rejection is not well founded.

Claims 2-8 depend from claim 1 and the remarks made above regarding claim 1 apply to these claims as well. In view of the foregoing, it is submitted that claim 1 and its dependents are allowable over Hauck in view of Greaves. In addition, it is noted that some of the dependent claims add limitations which are not shown or suggested by Hauck or Greaves. For example, claim 3 provides that the step of mapping one of the input interfaces to one of the tunnel interfaces is performed by using context data in an arriving packet as a first search key to a first database. The Examiner states that Hauck teaches using first packet data to map a flow of packets. The portion of Hauck cited by the Examiner actually teaches "on a first packet, a hash value (using label, QoS, data) resolves to a micro-flow which has not yet been created. Forwarding then takes the label and performs a lookup which results in an aggregate flow block. The micro-flowblock is then created and filled in by referring to the AFB. Subsequent packets then hash to the micro-flowblock which in turn refers to the AFB." There is no suggestion in Hauck to use context data in a packet to map an input interface to a one of a plurality of tunnel interfaces, each implementing a different tunneling protocol. Thus, the Examiner's rejection of claim 3 is not well founded.

As another example, claim 6 provides that both the step of mapping one of the input interfaces to one of said tunnel interfaces (from a set of tunnel interfaces which implement different tunneling protocols) and said step of mapping said one of said tunnel interfaces to one of the output interfaces are performed by using context data in an arriving packet as a first search key to a first database. The Examiner states that Hauck teaches claim 6 but the cited portion of Hauck only teaches routing microflows. Thus, the Examiner's rejection of claim 6 is not well founded.

In the Final Rejection of claim 15, the Examiner relies on Greaves as described above with respect to claim 1. Independent claim 15 was previously amended in a manner similar to the amendment to claim 1. Claim 15 clearly sets forth a method for implementing ***different tunneling protocols*** in a uniform manner with structural support in the body of the claim for the function recited in the preamble. In particular, claim 15 includes "mapping input streams and output streams to tunnel interfaces for different tunneling protocols in a uniform manner". As explained in detail above, Hauck describes multiple tunnels but does not describe or suggest how to implement different tunneling protocols and does not describe or suggest implementing multiple tunneling protocols in a uniform manner. Care must be taken not to confuse the phrase "multiple tunnels" with the phrase "multiple tunneling protocols". In view of the foregoing, it is submitted that claim 15 and its dependents are allowable over Hauck and Greaves for the same reasons presented above regarding claim 1.

Independent claim 22 was previously amended in a manner similar to the amendments made to claims 1 and 15. It is believed that claim 22 clearly sets forth a method for implementing different tunneling protocols in a uniform manner with structural support in the body of the claim for the function recited in the preamble. As explained in detail above, Hauck describes multiple tunnels but not describe or suggest implementing different tunneling protocols and does not describe or suggest implementing multiple tunneling protocols in a uniform manner. Greaves is not analogous art as explained above with regard to claim 1 and would not be used or understood by those skilled in the art to suggest implementing multiple tunneling protocols as claimed. In fact, as explained above, Greaves specifically teaches away from implementing protocols as claimed. In view of the foregoing, it is submitted that claim 22 and its dependents are allowable over Hauck in view of Greaves.

c) Claims 9 and 35 stand rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of Hauck.

Independent claim 9 is a method for implementing ***multiple tunneling protocols*** in a uniform manner which includes the step of associating an input interface, an output interface, and an information database with ***each of said multiple tunneling protocols***, associating a mapping interface and a mapping information base with ***each of said multiple tunneling protocols***, and uniformly implementing a tunneling protocol by selecting an input interface, an output interface, and an information database associated with the tunneling protocol to be implemented.

As explained above with reference to claim 31, there is no suggestion in Shrader of implementing a plurality of tunneling protocols. As explained above with reference to claim 1, Hauck neither teaches nor suggest a method for implementing multiple tunneling protocols. It is believed that the Examiner is improperly equating the phrase "multiple tunnels" with the phrase "multiple tunneling protocols". Thus, the obviousness rejection of claim 9 is not well founded.

Claim 35 depends from claim 31 which was argued with respect to Shrader. The Examiner has cited Hauck as teaching "multiple entry and exit node structures" and refers to Fig. 1A of Hauck. Fig. 1A is described by Hauck as follows: "FIG. 1A is an illustration of an exemplary segment of a micro-flow LSP network domain 100.... The micro-flow LSP domain 100 includes ingress LSR 102,..., and an egress LSR 110." Col. 7, lines 34 et seq. It is considered that the word "ingress" is equivalent to the claimed word "entry" and the claimed word "exit" is equivalent to the word "egress". Thus, Hauck specifically teaches a single entry and a single exit, whereas claim 35 claims a plurality of each. Therefore, the obviousness rejection of claim 35 is not well founded.

d) Claim 14 stands rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of Hauck and admitted prior art.

Claim 14 depends from claim 9 and the arguments made above regarding Claim 9 apply to claim 14 as well. The admitted prior art does not provide what was is lacking in

Shrader and Hauck, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 14 is allowable for the same reasons as claim 9.

<u>e) Claims 16-21, 23, and 28 stand rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves and further in view of Miller.</u>

Claim 16 depends from claim 15 and the arguments made above regarding claim 15 apply to claim 16 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 16 is allowable for the same reasons as claim 15. Moreover, the cited portion of Miller does not teach what the Examiner states it does.

Claim 17 depends from claim 16 and the arguments made above regarding claim 16 apply to claim 17 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router.. Thus, claim 17 is allowable for the same reasons as claim 16. Moreover, the cited portion of Miller does not teach what the Examiner states it does.

Claim 18 depends from claim 16 and the arguments made above regarding claim 16 apply to claim 18 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 18 is allowable for the same reasons as claim 16. Moreover, the cited portion of Miller does not teach what the Examiner states it does.

Claim 19 depends from claim 18 and the arguments made above regarding claim 18 apply to claim 19 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 19 is allowable for the same reasons as claim 18. Moreover, the cited portion of Miller does not teach what the Examiner states it does.

Claim 20 depends from claim 19 and the arguments made above regarding claim 19 apply to claim 20 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 20 is allowable for the same reasons as claim 19. Moreover, the cited portion of Miller does not teach what the Examiner states it does.

Claim 21 depends from claim 17 and the arguments made above regarding claim 17 apply to claim 21 as well. Miller does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 21 is allowable for the same reasons as claim 17. Moreover, the cited portion of Hauck does not teach what the Examiner states it does.

f) Claim 24 stands rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves in view of Miller and further in view of Rekhter.

Claim 24 depends from claim 23 which depends from claim 22 and the arguments made above regarding claim 22 apply to claim 24 as well. Neither Miller nor Rekhter provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 24 is allowable for the same reasons as claim 22.

g) Claim 26 stands rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves and further in view of Shrader.

Claim 26 depends from claim 22 and the arguments made above regarding claim 22 apply to claim 26 as well. Shrader does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 26 is allowable for the same reasons as claim 22.

h) Claim 27 stands rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves in view of Shrader and further in view of Tsirtsis.

Claim 27 depends from claim 26 which depends from claim 22 and the arguments made above regarding claim 22 apply to claim 27 as well. Neither Shrader nor Tsirtsis provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 27 is allowable for the same reasons as claim 22.

i) Claims 29 and 30 stand rejected under 35 U.S.C. §103(a) as obvious over Hauck in view of Greaves and further in view of admitted prior art.

Claim 29 depends from claim 22 and the arguments made above regarding claim 22 apply to claim 29 as well. The admitted prior art does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 29 is allowable for the same reasons as claim 22.

Claim 30 depends from claim 29 and the arguments made above regarding claim 29 apply to claim 30 as well. The admitted prior art does not provide what is lacking in Hauck and Greaves, i.e. implementing multiple tunneling protocols in a uniform manner in a single switch or router. Thus, claim 30 is allowable for the same reasons as claim 29.

j) Claims 33 and 38 stand rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of Miller.

Claim 33 depends from claim 31 and the arguments made above regarding claim 31 apply to claim 33 as well. Miller does not provide what is lacking in Shrader, i.e. implementing multiple tunneling protocols in a single switch or router. Thus, claim 33 is allowable for the same reasons as claim 31.

Claim 38 depends from claim 33 and the arguments made above regarding claim 33 apply to claim 38 as well. Thus, claim 38 is allowable for the same reasons as claim 33.

<u>k) Claim 34 stands rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of Miller and further in view of Rekhter.</u>

Claim 34 depends from claim 33 which depends from claim 31 and the arguments made above regarding claim 31 apply to claim 34 as well. Neither Miller nor Rekhter provide what is lacking in Shrader, i.e. implementing multiple tunneling protocols in a single switch or router. Thus, claim 34 is allowable for the same reasons as claim 31.

<u>l) Claim 37 stands rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of Tsirtsis.</u>

Claim 37 depends from claim 36 which depends from claim 31 and the arguments made above regarding claim 31 apply to claim 37 as well. Tsirtsis does not provide what is lacking in Shrader, i.e. implementing multiple tunneling protocols in a single switch or router. Thus, claim 37 is allowable for the same reasons as claim 31.

m) Claims 39 and 40 stand rejected under 35 U.S.C. §103(a) as obvious over Shrader in view of admitted prior art.

Claim 39 depends from claim 31 and the arguments made above regarding claim 31 apply to claim 39 as well. The admitted prior art does not provide what is lacking in Shrader, i.e. implementing multiple tunneling protocols in a single switch or router. Thus, claim 39 is allowable for the same reasons as claim 31.

Claim 40 depends from claim 39 and the arguments made above regarding claim 39 apply to claim 40 as well.

In light of all of the above, it is submitted that the claims are in order for allowance, and prompt allowance is earnestly requested. Should any issues remain outstanding, the Examiner is invited to call the undersigned attorney of record so that the case may proceed expeditiously to allowance.

Respectfully submitted,

/Thomas A Gallagher/

Thomas A Gallagher
Reg. No. 31,358
Attorney for Applicant(s)

GORDON & JACOBSON, P.C.
60 Long Ridge Road
Suite 407
Stamford, CT 06902
voice: (203) 323-1800
fax: (203) 323-1803

December 7, 2007

VIII. Claims Appendix

1. A uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input interfaces and a plurality of output interfaces, comprising:

a) providing a finite set of tunnel interfaces, each tunnel interface characterized by a set of tunnel specific attributes, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol;

b) mapping one of the input interfaces to one of said tunnel interfaces; and

c) mapping said one of said tunnel interfaces to one of the output interfaces, whereby

multiple tunneling protocols are implemented in a uniform way.

2. The method according to claim 1, wherein:

said tunnel specific attributes include parameters identifying tunnel end points.

3. The method according to claim 1, wherein:

said step of mapping one of the input interfaces to one of said tunnel interfaces is performed by using context data in an arriving packet as a first search key to a first database.

4. The method according to claim 3, wherein:

said arriving packet has a header and said context data is obtained from said header.

5. The method according to claim 4, further comprising:

d) processing said header with said one of said tunnel interfaces to obtain a new header, wherein said step of mapping said one of said tunnel interfaces to one of the output interfaces is performed by using the new header as a second search key to a second database.

6. The method according to claim 1, wherein:

both said step of mapping one of the input interfaces to one of said tunnel interfaces and said step of mapping said one of said tunnel interfaces to one of the output interfaces are performed by using context data in an arriving packet as a first search key to a first database.

7. The method according to claim 6, wherein:

said arriving packet has a header and said context data is obtained from said header.

8. The method according to claim 5, wherein:

the one of the output interfaces is one of an L2 (layer two) and an L3 (layer three) interface, and said step of using the new header as a second search key to a second database yields one of an L2 and an L3 interface.

9. A uniform method for implementing multiple tunneling protocols in a switch or router, comprising:

a) associating an input interface, an output interface, and an information database with each of said multiple tunneling protocols;

b) associating a mapping interface and a mapping information base with each of said multiple tunneling protocols; and

c) uniformly implementing a tunneling protocol by selecting an input interface, an output interface, and an information database associated with the tunneling protocol to be implemented.

14. The method according to claim 9, wherein:

for ETHERNET over MPLS (multiprotocol label switching) tunnel origination, the input interface is an ETHERNET interface, the output interface is an L2 (layer 2) interface, and the information database is a switching information base.

15. A uniform method for implementing multiple tunneling protocols in a switch or router having a plurality of input streams and a plurality of output streams, comprising:

a) providing a finite set of tunnel interfaces, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol; and

b) mapping input streams and output streams to tunnel interfaces for different tunneling protocols in a uniform manner.

16. The method according to claim 15, wherein:

some of the input streams are L2 (layer two) streams and some of the input streams are L3 (layer 3) streams, said step of providing a finite set of tunnel interfaces includes providing a set of L2 tunnel interfaces for L2 input streams and a set of L3 tunnel interfaces for L3 input streams.

17. The method according to claim 16, wherein:

input streams are mapped to tunnel interfaces by a forwarding function.

18. The method according to claim 16, wherein:

L2 input streams are mapped to L2 tunnel interfaces by a first forwarding function, and L3 input streams are mapped to L3 tunnel interfaces by a second forwarding function.

19. The method according to claim 18, wherein:

some of the output streams are L2 streams and some of the output streams are L3 streams, L2 tunnel interfaces are mapped to L2 output streams by a third forwarding function, and L3 tunnel interfaces are mapped to L3 output streams by a fourth forwarding function.

20. The method according to claim 19, wherein:

L2 tunnel interfaces are mapped to L3 output streams by a fifth forwarding function, and L3 tunnel interfaces are mapped to L2 output streams by a sixth forwarding function.

21. The method according to claim 17, wherein:

the forwarding function performs mapping based on context data associated with input packets and database information which is configured and updated by a local host.

22. A uniform method for implementing multiple tunneling protocols in a switch or router, comprising:

providing a plurality of tunnel interfaces, a first of said tunnel interfaces being associated with one tunneling protocol and a second of said tunnel interfaces being associated with a second tunneling protocol different from said first tunneling protocol, each tunnel interface having a plurality of parameters which are described in a uniform way, said plurality of parameters including a local source address and a remote destination address.

23. The method according to claim 22, wherein:

said plurality of parameters includes a hop limit or time to live.

24. The method according to claim 23, wherein:

said plurality of parameters includes a tunnel MTU (maximum transmission unit).

25. The method according to claim 22, further comprising:

providing a plurality of tunnel entry node structures and a plurality of tunnel exit node structures

26. The method according to claim 22, further comprising:

providing an address function to set tunnel interface source and destination addresses.

27. The method according to claim 26, further comprising:

providing a first address function for IPv4 (internet protocol version four) interfaces and a second address function for IPv6 (internet protocol version six) interfaces.

28. The method according to claim 23, further comprising:

providing a hop function to set the hop limit for a tunnel interface.

29. The method according to claim 22, wherein:

said plurality of parameters includes MPLS (multiprotocol label switching) encapsulation information and actions to be performed on MPLS packets.

30. The method according to claim 29, further comprising:

providing an MPLS function to associate an MPLS LIB (label information base) with an MPLS interface.

31. An application programming interface (API) for implementing a plurality of different tunneling protocols in a switch or router, said API comprising:

a) a tunneling interface data structure having a plurality of parameters; and

b) a plurality of functions for setting the parameters of the tunneling interface data structure, wherein

a tunneling interface data structure is configurable to implement any one of said plurality of different tunneling protocols by using at least some of said plurality of functions.

32. The API according to claim 31, wherein:

said plurality of parameters including a local source address and a remote destination address.

33. The API according to claim 32, wherein:

said plurality of parameters includes a hop limit or time to live.

34. The API according to claim 33, wherein:

said plurality of parameters includes a tunnel MTU (maximum transmission unit).

35. The API according to claim 31, further comprising:

c) a plurality of tunnel entry node structures; and

d) a plurality of tunnel exit node structures.

36. The API according to claim 31, wherein:

said plurality of functions includes an address function to set tunnel interface source and destination addresses.

37. The API according to claim 36, wherein:

said plurality of functions includes a first address function for IPv4 (internet protocol version four) interfaces and a second address function for IPv6 (internet protocol version six) interfaces.

38. The API according to claim 33, wherein:

said plurality of functions includes a hop function to set the hop limit for a tunnel interface.

39. The API according to claim 31, wherein:

said plurality of parameters includes MPLS (multiprotocol label switching) encapsulation information and actions to be performed on MPLS packets.

40. The API according to claim 39, wherein:

said plurality of functions includes an MPLS function to associate an MPLS LIB (label information base) with an MPLS interface.

IX. Evidence Appendix

The following six pages represent a pseudocode appendix that was submitted as an ASCII text file on a CDROM with the application. The lines are numbered consecutively and are referred to in the specification as filed as well as in this Brief.

```
/*
** This header file defines typedefs, constants, and functions
** that apply to the Transwitch Tunnel Interface Management API.
*/

#ifndef __TXC_TUNNEL_H_
#define __TXC_TUNNEL_H_

#ifdef __cplusplus
extern "C"
#endif

/*
** Transwitch definitions
*/
typedef char                                    TXC_char8_t;
typedef unsigned char                 TXC_uchar8_t;
typedef char                                    TXC_int8_t;
typedef short                                   TXC_int16_t;
typedef int                                     TXC_int32_t;
typedef long long int                 TXC_int64_t;
typedef unsigned char                 TXC_uint8_t;
typedef unsigned short                TXC_uint16_t;
typedef unsigned int                       TXC_uint32_t;
typedef unsigned long long long long int  TXC_uint128_t;

/*
** API function call return code
*/
typedef TXC_uint32_t   TXC_error_t;


/*
 * Interface ID
 */
typedef TXC_uint32_t   TXC_IfID_t;

/*
** Information Base (FIB, LIB, TSIB, SIB) Identifier
*/
typedef TXC_uint32_t   TXC_InfoBaseID_t;


/*
** IPv4 and IPv6 Addresses
*/

typedef TXC_uint32_t                TXC_IPv4Address_t;
typedef TXC_unit128_t         TXC_IPv6Address_t;

/*
** MPLS Label Stack
*/

typedef struct {
TXC_int32_t numLabels; /* Number of labels */
TXC_int32_t *labelStack; /* Stack of labels */
} TXC_MPLS_LabelStack_t;


/*
** Tunnel Interface Attributes
*/

      typdef struct {

             TXC_IfType_t     Interface_type;

             union {
TXC_IfTunnelIPv4_t        TunnelIPv4; /*IPv4 in IPv4 */
      TXC_IfTunnelIPv6_t        TunnelIPv6; /*IPv6 in IPv6 */
                  TXC_IfTunnelIPMpls_t      TunnelIMPLS;/* IP over MPLS*/
                  TXC_IfTunnelEthernetMPLS_t TunnelEMPLS;/* Ethernet/MPLS */

                  } u;
}If_TunnelGeneric_t;


/*
**    IPv4 Tunnel Interface attributes.
*/
typedef struct {
uint8_t     TunnelProtocol; /* IPv4 in IPv4 Protocol */

uint16_t mtu;             /* Tunnel Max Transmission Unit */
              /* IPv4 MTU - IPv4 header size*/
uint8_t NumberHops;       /* IPv4 Tunnel Number of Hops*/

IPv4Addr_t  Tunnel_IPv4SrcAddr;
 /* Tunnel Entry Node Address */
IPv4Addr_t  Tunnel_IPv4DstAddr;
/* Tunnel Entry Node Address *

} TXC_IfTunnelIPv4_t;

/*
**    IPv6 Tunnel Interface attributes.
*/
typedef struct {
TXC_uint8_t       TunnelProtocol; /* IPv6 in IPv6 Protocol */

TXC_uint16_t      mtu;        /* Tunnel Max Transmission Unit */
                  /* IPv6 MTU - IPv6 header size*     TXC_uint8_t
      NumberHops; /* IPv6 Tunnel Number of Hops*/


TXC_IPv6Addr_t    Tunnel_IPv6SrcAddr;
 /* Tunnel Entry Node Address */
TXC_IPv6Addr_t    Tunnel_IPv6DstAddr;
/* Tunnel Entry Node Address */

} TXC_IfTunnelIPv6_t;

/*
```

```
**    MPLS Tunnel Interface attributes.
*/
typedef struct {

TXC_uint16_t      mtu;         /* Tunnel Max Transmission Unit */
                  /* MTU - MPLS header size*    TXC_uint8_t       NumberHops;
      /* MPLS Tunnel Number of Hops*/

TXC_MPLS_LabelStack_t LabelStack;
/* MPLS Label Stack */

} TXC_IfTunnelMPLS



/*
**    L2TP Tunnel Interface attributes.
*/
typedef struct {

TXC_uint32_t      helloInterval;    /* Hello Interval */
TXC_uint32_t      idleTimeout;      /* Idle Timeout */
TXC_uint32_t      receiveWindowSize;/* ReceiveWindowSize */
   TXC_uint32_t   retransMax;       /* Max retrans */
      TXC_uint32_t      retransTimeoutMax;/* Max retrans timout*/
TXC_uint32_t      reasTimeout;      /* reassembly timeout */

union {
      TXC_IfTunnelIPv4_t  Ipv4If;
TXC_IfTunnelIPv6_t  Ipv6If;
      }
} TXC_IfTunnelL2TP


/*
** Tunnel Interface Types
*/

typedef enum {
  TXC_IF_TYPE_IPV4Tunnel=1,    /* IPv4 Tunnel interface */
  TXC_IF_TYPE_IPV6Tunnel=2,    /* IPv6 Tunnel interface */
  TXC_IF_TYPE_IPMPLSTunnel=3, /* IP/MPLS Tunnel interface */
  TXC_IF_TYPE_EthMPLSTunnel=4 /* Ethernet/MPLS Tunnel interface*/
} TXC_IfType_t;

/*
** IPv4 Tunnel Source and Destination Address definitions.
*/
typedef struct {
TXC_IPv4Addr_t    IPv4Tunnel_SrcAddr;
TXC_IPv4v4Addr_t IPv4Tunnel_DstAddr;
} TXC_IPv4TunnelAddr_t ;

/*
** IPv6 Tunnel Source and Destination Address definitions.
*/
typedef struct {
```

```
TXC_IPv6Addr_t    IPv6Tunnel_SrcAddr;
TXC_Ipv6v4Addr_t IPv6Tunnel_DstAddr;
} TXC_IPv6TunnelAddr_t ;

typdef struct {
      union {
      TXC_IPv4TunnelAddr_t IPv4AddrPair;
      TXC_IPv6TunnelAddr_t IPv6AddrPair;
            }u;
} TXC_IPTunnelAddr_t ;

/*
** Function to set an IP (IPv4, or IPv6) Tunnel Source and Destination **
Nodes Address
*/
TXC_error_t TXC_IfTunnelIPv4AddrSet(
                  TXC_uint32_t            n_interfaces,
                  TXC_IfID_t              *if_IDArray,
                  TXC_IPTunnelAddr_t      *if_IPTunnelAddrArray);


/*
** Function to set MPLS Tunnel Label Stacks on a set of Interfaces
*/
TXC_error_t TXC_IfTunnelMPLSSet(
             TXC_uint32_t           n_interfaces,
             TXC_IfID_t             *if_IDArray,
             TXC_MPLS_LabelStack_t  *if_MPLSLabelStackArray);



/*
** Function to set the Number of Hops of a Tunnel
*/
TXC_error_t TXC_IfTunnelNumberHopsSeetSet(

             TXC_uint32_t n_interfaces,
             TXC_IfID_t *if_IDArray,
             TXC_uint8_t *if_TunnelNumberHopsArray);

/*
** Function to associate an IP FIB, an MPLS LIB, TSIB or SIB
 */

TXC_error_t TXC_IfInfoBaseSet(
 TXC_uint32_t          n_interfaces,
 TXC_IfType_t       *if_TypeArray,
 TXC_IfID_t         *if_IDArray,
 TXC_Info_ID_t      *if_Info_IDArray);




/*
** Function to set L2TP interface attributes
 */

```

```
TXC_error_t TXC_IfL2TPAttrSet(
TXC_IfID_t     if_ID,
TXC_uint32_t      helloInterval,
TXC_uint32_t      idleTimeout,
TXC_uint32_t      receiveWindowSize,
      TXC_uint32_t      retransMax,
            TXC_uint32_t      retransTimeoutMax,
TXC_uint32_t      reasTimeout);


/*
 *    Error codes returned to function calls invocations
 */

#define TXC_TUNNEL_BASE_ERR=256

/* Invalid parameter */
#define TXC_IF_E_INVALID_PARAM ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+1)

/* Invalid Interface Type */
#define TXC_IF_E_INVALID_IF_TYPE ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+2)

/* Array length <= 0 or too big */
#define TXC_IF_E_BAD_ARRAY_LENGTH ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+3)

/* Invalid MTU specification */
#define TXC_IF_E_INVALID_MTU ((TXC_IfErrorType_t)
TXC_TUNNEL_BASE_ERR+4)

/* Invalid FIB ID */
#define TXC_IF_E_INVALID_FIB_ID ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+5)

/* Invalid LIB ID */
#define TXC_IF_E_INVALID_LIB_ID ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+6)

/* Invalid TSIB ID */
#define TXC_IF_E_INVALID_TSIB_ID ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+7)

/* Invalid SIB ID */
#define TXC_IF_E_INVALID_SIB_ID ((TXC_IfErrorType_t) TXC_TUNNEL_BASE_ERR+8)

/* Invalid Number of Hops value */
#define TXC_IF_E_INVALID_NUMBER_HOPS((TXC_IfErrorType_t)
TXC_TUNNEL_BASE_ERR+9)

/* Invalid Tunnel Source Address */
#define TXC_IF_E_INVALID_TUNNEL_SRCADDR ((TXC_IfErrorType_t)
TXC_TUNNEL_BASE_ERR+10)

/* Invalid Tunnel Destination Address */
#define TXC_IF_E_INVALID_TUNNEL_DSTADDR ((TXC_IfErrorType_t)
TXC_TUNNEL_BASE_ERR+11)

/* Invalid L2TP Tunnel Interface Attribute */
#define TXC_IF_E_INVALID_L2TP_ATTR ((TXC_IfErrorType_t)
TXC_TUNNEL_BASE_ERR+12)

```

```
#ifdef __cplusplus
}
#endif

#endif /* __TXC_TUNNEL_H_ */


Tunnel API        Transwitch Corporation
```

X. Related Proceedings Appendix

None.